Dense Map Inference with User-Defined Priors: From Priorlets to Scan Eigenvariations

Paloma de la Puente[1,⋆] and Andrea Censi[2]

[1] Intelligent Control Group, Universidad Politecnica de Madrid, C/ Jose Gutierrez Abascal, 2. 28006 Madrid, Spain
paloma.delapuente@upm.es
[2] Engineering and Applied Science, California Institute of Technology, Pasadena, MC 107-81, 91125, CA, USA
andrea@cds.caltech.edu

Abstract. When mapping is formulated in a Bayesian framework, the need of specifying a prior for the environment arises naturally. However, so far, the use of a particular structure prior has been coupled to working with a particular representation. We describe a system that supports inference with multiple priors while keeping the same dense representation. The priors are rigorously described by the user in a domain-specific language. Even though we work very close to the measurement space, we are able to represent structure constraints with the same expressivity as methods based on geometric primitives. This approach allows the intrinsic degrees of freedom of the environment's shape to be recovered. Experiments with simulated and real data sets will be presented.

1 Introduction

The Simultaneous Localization and Mapping (SLAM) problem is usually formulated in a Bayesian framework [16]. This paper concerns the use of prior distributions for the map: how to rigorously specify them and how to create an inference engine that works with multiple user-defined priors.

To see what role the prior plays in the problem, let us introduce some notation. Let $\boldsymbol{q}$ be the robot pose, let $\boldsymbol{m}$ be a variable representing the map, and let $\boldsymbol{z}$ be the measurements (including odometry and exteroceptive sensors), which follow the known sensor model $p(\boldsymbol{z}|\boldsymbol{q}, \boldsymbol{m})$. SLAM can be formulated as the problem of estimating $p(\boldsymbol{q}, \boldsymbol{m}|\boldsymbol{z})$, the joint distribution of pose and map conditioned to the measurements. We focus on the case of mapping with dense sensors and maps; if the map consists of landmarks, then most of the following remarks are not relevant. More specifically, we describe the formulation that uses the Rao-blackwellization technique [4], where one approximates the target distribution as $p(\boldsymbol{q}, \boldsymbol{m}|\boldsymbol{z}) \simeq p(\boldsymbol{q}|\boldsymbol{z})p(\boldsymbol{m}|\boldsymbol{q}, \boldsymbol{z})$, thereby factorizing SLAM in two subproblems: estimating the pose of the robot given the measurements ($p(\boldsymbol{q}|\boldsymbol{z})$), and mapping

⋆ Paloma de la Puente's work was supported in part by Comunidad de Madrid and European Social Fund (ESF) programs and also by the Spanish Ministry of Science and Technology, DPI2010-21247-C02-01 - ARABOT (Autonomous Rational Robots).

with known poses ($p(\boldsymbol{m}|\boldsymbol{q},\boldsymbol{z})$). Let us focus on the latter. Given the sensor model, we can compute the posterior using Bayes' theorem: $p(\boldsymbol{m}|\boldsymbol{q},\boldsymbol{z}) \propto p(\boldsymbol{z}|\boldsymbol{q},\boldsymbol{m})p(\boldsymbol{m})$.

Therefore, if we want to compute the posterior distribution of the map $\boldsymbol{m}$ given the observations, we need to know the prior $p(\boldsymbol{m})$. We remark that, had we formulated SLAM as a maximum-likelihood problem (find $\boldsymbol{m}$ that maximizes $p(\boldsymbol{z}|\boldsymbol{q},\boldsymbol{m})$), the knowledge of $p(\boldsymbol{m})$ would not be strictly necessary. That, however, would only work for finite-dimensional problems. In fact, if the underlying map is an arbitrary surface, the maximum-likelihood problem is ill posed, because the solution is any curve that perfectly interpolates the readings. To obtain a more reasonable solution, we always need some kind of regularization, which is the prior. Therefore, we conclude that, to make the SLAM problem with dense sensors and maps well posed, we have to specify a prior $p(\boldsymbol{m})$.

Other than to make the mathematical formulation correct, the knowledge of the prior helps in reducing the uncertainty of the estimate. For example, constraints such as collinearity are very powerful in reducing the map uncertainty. In general, any assumption about the environment that the user can provide helps in making the filter more efficient. Yet, to our knowledge, incorporating generic prior information in filters has never been done before, and that can be attributed to the representation used, which generally presents some limitations.

For instance, let us consider SLAM methods that represent maps using occupancy/evidence grids. Firstly, the grid resolution introduces some kind of spatial regularization, and makes it impossible to represent precise geometric primitives such as line segments. The other limitation is that each cell is assumed to be independent: this makes it impossible to effectively use the prior information because geometric constraints between different parts of the environment result in long-range correlation of cells occupancy.

A popular alternative to occupancy grids is using a map composed of geometric primitives (segments, circles, splines, etc.). In that case, the prior is implicit in the representation: representing a map by segments automatically gives non-segments map a zero prior. Using geometric primitives presents two major advantages: they provide explicit information about the geometrical nature of the environment, and the resulting maps are much more compact. With proper bookkeeping, the correlation between different parts of the environment can be precisely represented. However, they lack in flexibility. For example, in most realistic environments —except perhaps completely engineered factory floors— there will be parts of the environment that cannot be described by the prior. Moreover, often one wishes to impose "soft constraints": for example, rather than imposing that all walls are exact line segments, probably a better prior is that they are likely to be straight, or that they are of a bounded variation from straight; all these details should be figured out by the user. This flexibility cannot be accommodated by existing feature-based methods.

1.1 Contribution

We began this work by asking the question of whether it is possible to decouple the concept of prior from a particular representation. Instead of the prior being

hidden in the representation, can it be made completely explicit and under the direct control of the user? Can we have an inference engine that works with multiple priors?

In Section 2, we start by defining a new representation. A range-finder provides an array of numbers measuring the distance to the obstacles. We augment that by associating to each measurement the corresponding surface normal. This gives us the flexibility of occupancy grids with the precision of geometric primitives. Just like large environments can be represented by a collection of patches, one can represent any environment by a collection of augmented scans; still, in this paper, we focus on processing the data from a single scan, with an ego-centric perspective.

In our system, the prior is entirely provided by the user, who describes the structure constraints and the model likelihood as a function of readings and normals, in a particular domain-specific language (Fig. 2 on page 99).

The inference engine, described in Section 3, takes two inputs: the noisy raw distance readings from a laser sensor, and the user-specified prior. The output is the posterior distribution for the local map, represented as a Gaussian distribution on the space of readings and normals. This is obtained by a two-step process. In the first step, we solve a nonlinear optimization problem to obtain the mode of the distribution. In the second step, described in Section 4, we use the knowledge of the structure constraints to shrink the measurements covariance, by projecting it onto the allowed submanifold. Fig 1 shows a geometric interpretation of the process.

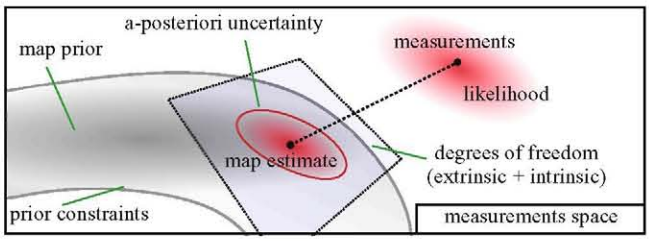


Fig. 1. Computing the posterior distribution of the map in the measurement space has a clear geometric interpretation. The prior $p(\boldsymbol{m})$ defines a (thin) surface in the measurement space. The initial measurements define a thick ellipse of uncertainty that gets projected and constrained to the prior surface.

We believe our method presents a novel approach for data segmentation and preprocessing in a flexible manner, being able to reduce the uncertainty of noisy measurements and providing information about the environment's geometrical nature. Section 5 includes some experiments showing how it works. In our opinion the proposed framework has a lot of potential for future research: it may also be very useful for scan-matching techniques with laser data and it is very promising for overall map optimization as well, helping us build better models of the physical world with a unique system in different situations.

1.2 Related Work

So far, prior information about the environment has been used explicitly only in feature-based SLAM methods. For example, Chong and Kleeman [3] employ collinearity constraints to enhance the state estimation with a Julier-Uhlman Kalman Filter. Rodríguez-Losada *et al.* [14] alleviate the inconsistency due to the linearization errors introduced by the Extended Kalman Filter by enforcing parallelism or orthogonality constraints. Parsley and Julier [12] propose a framework for the integration of prior information coming from different sources to improve the quality of feature based SLAM. Nguyen *et al.* [11] apply orthogonality constraints to build accurate simplified plane based 3D maps. Beevers and Huang [1] show that imposing a-priori known relative constraints also leads to consistency and efficiency improvements for particle filters. Other recent contributions are based on the graph-SLAM approach [7, 17]. In all these works, a particular geometrical model for representation is used, and they only support equality constraints.

We know of no previous work using a dense representation and allowing for the use of different priors provided by the user. Modelling the scans as a Gaussian process [13] does allow to impose a prior distribution, corresponding to a smoothness constraint, but it cannot capture structured priors such as polygonal environments.

1.3 Notation

Let $\boldsymbol{q} = (\boldsymbol{t}, \theta) \in \mathrm{SE}(2)$ be the robot pose. Assume, without loss of generality, that the range-sensor frame coincides with the robot frame. The sensor model for the range-sensor measurements $\tilde{\boldsymbol{\rho}} = \{\tilde{\rho}_i\}_{i=1}^n$ is defined by $\tilde{\rho}_i = \rho_i + \epsilon_i$, where ρ_i is the true distance to the obstacle, and ϵ_i is additive Gaussian noise with covariance $\Sigma_{ij} = \mathrm{cov}\{\epsilon_i, \epsilon_j\}$, not necessarily diagonal. The true distance to the obstacle can be written as

$$\rho_i = r(\boldsymbol{m}, \langle \boldsymbol{t}, \theta + \phi_i \rangle), \tag{1}$$

where the angle ϕ_i is the direction of each reading in the scan, and the function $r : \mathcal{M} \times \mathrm{SE}(2) \to \mathbb{R}_+ \cup \{\infty\}$ is the "ray-tracing" function that returns the distance to the closest obstacle from a certain pose. The function r depends on the map $\boldsymbol{m} \in \mathcal{M}$. For now, we do not specify anything about $\boldsymbol{m}$, just that it represents the underlying map of the environment.

1.4 Problem Statement

Formally, we divide the problem of approximating $p(\boldsymbol{m}|\boldsymbol{z}, \boldsymbol{q})$, where $\boldsymbol{z} = \tilde{\boldsymbol{\rho}}$, in two sub-problems. First, we solve the maximum-a-posteriori problem to obtain the mode of the distribution. Since $p(\boldsymbol{m}|\boldsymbol{z}, \boldsymbol{q}) \propto p(\boldsymbol{z}|\boldsymbol{m}, \boldsymbol{q})p(\boldsymbol{m})$, this can be posed as follows:

Problem 1. Find $\boldsymbol{m}$ that maximizes

$$\log p(\boldsymbol{z}|\boldsymbol{m}, \boldsymbol{q}) + \log p(\boldsymbol{m}).$$

The first term is simply the measurements likelihood; the second term is the map prior. After we have found the mode of the distribution, we obtain a Gaussian approximation to $p(\boldsymbol{m}|\boldsymbol{z}, \boldsymbol{q})$ by projecting the initial covariance onto the prior constraints (Fig. 1).

This process is conducted in a representation space very close to the measurement space, as described in the next section.

2 Defining Map Priors with Priorlets

The environment prior is specified by the user in a domain-specific language; a representative set of user-supplied prior definition files is shown in Fig. 2. Providing a flexible way to parametrize environment priors posed two challenges. The first mathematical challenge is choosing a unified representation that allows for the description of a multitude of priors. The second challenge is that this representation must also be user-friendly.

2.1 Representation: Distances $\boldsymbol{\rho}$, Normals $\boldsymbol{\alpha}$, Topology $\boldsymbol{\mathcal{T}}$

For what concerns the representation, our solution is parametrizing $p(\boldsymbol{m})$ by three finite-dimensional quantities.

True distance to the obstacle $\boldsymbol{\rho}$: The quantities $\{\rho_i\}_{i=1}^n$ were already defined as part of the sensor model in equation (1). They represent a zeroth-order approximation of the environment shape.

Surface normals $\boldsymbol{\alpha}$: The surface normals represent a first-order approximation of the environment shape, and will play an important role in defining the priors. The surface normal α_i can be written similarly to ρ_i as a function of the derivative of the ray-tracing function[1]. We define $\boldsymbol{x} \triangleq (\boldsymbol{\rho}, \boldsymbol{\alpha})$ and we write compactly:

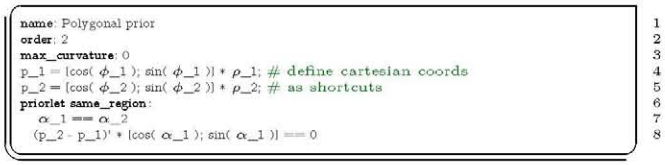
$$\boldsymbol{x} = (\boldsymbol{\rho}, \boldsymbol{\alpha}) = \boldsymbol{r}(\boldsymbol{m}, \boldsymbol{q}). \tag{2}$$

Environment topology $\boldsymbol{\mathcal{T}}$: We assume that the environment is partitioned into *surfaces,* and each *surface* is partitioned into one or more *regions.* For each two consecutive readings in the scan, there are three possible topology cases:

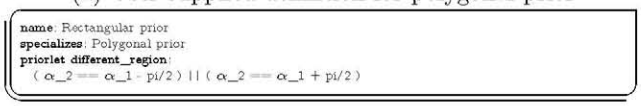
1. They belong to the same *surface* and the same *region.*
2. They belong to the same *surface*, but different *regions.*
3. They belong to different *surfaces.*

Having this fine distinction allows to precisely define the prior's constraints. To keep track of the topology information, we define a variable $\boldsymbol{\mathcal{T}} = \{\mathcal{T}_k\}_{k=1}^n$, where each $\mathcal{T}_k \in \{\texttt{sameRegion}, \texttt{differentRegion}, \texttt{differentSurface}\}$ describes the relation between a pair of consecutive points.

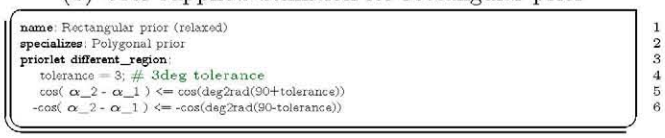
[1] An explicit expression for the normal α_i as a function of the ray-tracing function r is $\alpha_i = \pi/2 + \arctan\left(\frac{\partial}{\partial \phi_i} r\left(\boldsymbol{m}, \langle \boldsymbol{t}, \theta + \phi_i \rangle\right)\right)$, but we are not going to need it in this paper.



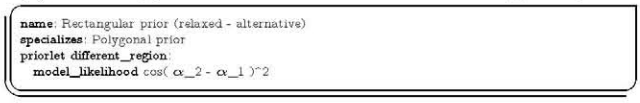
(a) User-supplied definition for polygonal prior



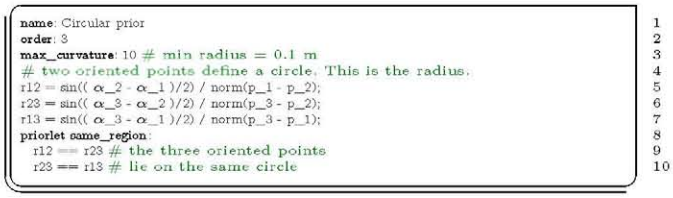
(b) User-supplied definition for rectangular prior



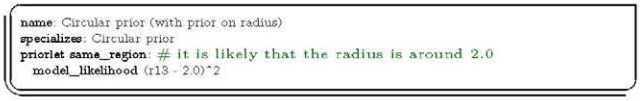
(c) User-supplied definition for relaxed rectangular prior



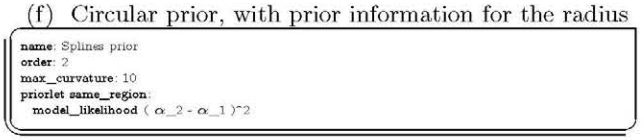
(d) User-supplied definition for alternative relaxed rectangular prior



(e) User-supplied definition for circular prior



(f) Circular prior, with prior information for the radius



(g) User-supplied definition for spline prior

Fig. 2. The environment prior is specified by the user with a domain-specific language. These are examples of actual source code interpreted by the inference engine (apart from some omissions in the interest of clarity). Using UNICODE, the special variables `alpha_i`, `rho_i`, `phi_i` can also be typed with Greek letters; this was inspired by Sun's Fortress language.

2.2 Expressing Priors as Functions of $\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}}$

We can express the prior as a function of the readings $\boldsymbol{\rho}$, normals $\boldsymbol{\alpha}$, and topology $\boldsymbol{\mathcal{T}}$ instead of as a function of the infinite-dimensional map $\boldsymbol{m}$. Assuming that it is possible, we rewrite Problem 1 as follows.

Problem 2. Find $\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}}$ that maximize

$$\log p(\tilde{\boldsymbol{\rho}}|\boldsymbol{\rho}) + \log p(\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}}).$$

Now we are dealing with a finite-dimensional optimization problem: the infinite-dimensional map "$\boldsymbol{m}$" has disappeared from the formalization. The limitation is that we can only define shape priors by their 0th (ρ) and 1st order (α) Taylor expansions. In the same spirit, we could use successive derivatives (curvature, and so on); nevertheless, we found that this parametrization has good expressivity. This does not mean that we are limited to piece-wise linear shapes; in fact, we can define shapes such as circles (Fig. 2e) and splines (Fig. 2g).

2.3 Expressing $p(\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}})$ with Local Constraints and Energies

Now we have fixed the representation, but we still have to solve the challenge of allowing the user to specify a prior in an intuitive way. It is clear that we can express almost any shape using a function $p(\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}})$. In theory, we could ask the user to provide a symbolic expression for $p(\boldsymbol{\rho}, \boldsymbol{\alpha}, \boldsymbol{\mathcal{T}})$. This, however, would be burdensome: assuming, for example, that there are 180 readings in a scan, the user would need to provide a symbolic expression with 540 variables. Moreover, that expression would have to be changed if the number of readings changed.

Our observation was that one can define interesting priors by describing *local* constraints between consecutive points. For example, if the environment prior is polygonal, we want to impose that nearby points have the same normals if they belong to the same region: $\alpha_1 = \alpha_2 = \cdots = \alpha_n$. This can be expressed compactly by saying that $\alpha_i = \alpha_{i+1}$ if points i and $i+1$ belong to the same region (compare Fig. 2a, line 7). In addition to these, we need constraints on ρ_i to ensure that the points are aligned (Fig. 2a, line 8).

In the case of a rectangular prior, we have the additional constraint that $(\alpha_i - \alpha_{i+1}) = k\frac{\pi}{2}$ if the two points do *not* belong to the same region (see Fig. 2b, line 4). Similarly, one can define different relaxations for a rectangular prior (Fig. 2c-2d). We will not describe in detail the interpretation of all the expressions in Fig. 2, but they all correspond to simple geometric constraints.

Certain priors cannot be specified by considering only two successive points. For example, it takes three consecutive points to describe a circular prior (see Fig. 2e), because it takes three points to define a circle. The *order* of a prior is the number of consecutive points needed for describing it.

2.4 Formal Definitions of Priorlets

We use the term *priorlet* for a set of local constraints plus energies imposed on n consecutive points in the environment.

Definition 1. *A* priorlet *of order n is a tuple $\langle F, G, H\rangle$ described by three sets of functions $F = \{f_k\}$, $G = \{g_k\}$, $H = \{h_k\}$. The arguments of all these functions are n couples of (distance, normal angle) and they all return a scalar. The functions $\{f_k\}$ represent equality constraints, the functions $\{g_k\}$ represent inequality constraints, and the functions $\{h_k\}$ represent "energies" (negative log-likelihoods).*

The semantics of a priorlet is the specification of a small part of a larger optimization problem:

$$\min_{\rho_{1:n},\alpha_{1:n}} \quad \ldots + \textstyle\sum_k h_k((\rho_1,\alpha_1),\cdots,(\rho_n,\alpha_n)) + \ldots \ ,$$

$$\begin{aligned} \textit{subject to} \qquad & f_k((\rho_1,\alpha_1),\cdots,(\rho_n,\alpha_n)) = 0, \\ & g_k((\rho_1,\alpha_1),\cdots,(\rho_n,\alpha_n)) \leq 0. \end{aligned}$$

The philosophy is very close to that of factor graphs *[6]; the formalization, however, does not match perfectly because usually factor graphs do not include constraints.*

Definition 2. *A* user-defined environment prior *is a collection of three priorlets: a "`same_region`" priorlet, a "`different_region`" priorlet, and a "`different_surface`" priorlet, describing the constraints/energies for neighbouring points for the three topology cases.*

Recall that the variable $\boldsymbol{\mathcal{T}}$ specifies the environment partition in regions and surfaces. Given a particular choice of $\boldsymbol{\mathcal{T}}$, we know which priorlet to apply to each couple (or triplet) of consecutive points. Therefore, we can define three functions $h_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha})$, $f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha})$, $g_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha})$. These represent, respectively, the cumulative effect of all the energies, and the stacked equalities and inequalities given by the application of the priorlets to each neighbourhood of points (we do not write them explicitly to avoid drowning in a sea of indices). We can rewrite Problem 2 as follows.

Problem 3. Find $\boldsymbol{\mathcal{T}}, \boldsymbol{\rho}, \boldsymbol{\alpha}$ as the solution of the problem:

$$\max_{\boldsymbol{\mathcal{T}},\boldsymbol{\rho},\boldsymbol{\alpha}} \quad \log p(\tilde{\boldsymbol{\rho}}|\boldsymbol{\rho}) + h_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha}),$$

$$\begin{aligned} \text{subject to} \qquad & f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha}) = \mathbf{0}, \\ & g_{\boldsymbol{\mathcal{T}}}(\boldsymbol{\rho},\boldsymbol{\alpha}) \leq \mathbf{0}. \end{aligned}$$

2.5 A Domain-Specific Language for Priorlets

We have given a formal description of priorlets that might appear overly complicated. In practice, the process of specifying a prior is intuitive, using a domain-specific language whose syntax we believe easy to understand even without a formal definition.

The user must minimally specify a `name`, and the `order` of the prior. Then she specifies the three priorlets (a `same_region` priorlet, a `different_region` priorlet, and a `different_surface` priorlet), by specifying equalities (`==`) and inequalities (`<=`) over the predefined variables `rho_`i, `alpha_`i, `phi_`i, with $1 \leq i \leq$ order. Using UNICODE input, the variables can also be indicated with Greek letters. At any point in the file, other variables can be introduced using "`=`" (Fig. 2a, line 4). The syntax for the expressions is the one used by MATLAB/Octave. A "`||`" operand is supported for specifying a logical or (Fig. 2b, line 4). The model likelihood (the h function) is introduced by the keyword `model_likelihood`. The user can subclass existing priors using the `specializes` keyword; for example, the rectangular prior specializes the polygonal prior (Fig. 2b, line 2). Finally, we let the user specify an explicit `max_curvature` parameter that is used in the inference process.

3 Inference with Generic Priors

Our goal has been to build an inference engine that works for arbitrary user-specified priors. Of course, we are doomed to be less efficient than an optimization method designed for a particular prior; however, we believe there is value in showing a completely general approach. In this section, we briefly recall the standard constrained-optimization methods that we use, we show how additional constraints can be added to the problem, and finally we discuss the two-level optimization procedure.

3.1 Homotopy Methods

The idea of homotopy methods [15] is to solve the constrained optimization problem by solving a *sequence* of *unconstrained* optimization problems. The *penalty function* method is useful for dealing with equalities or inequalities. Suppose the minimization problem to solve is

$$\min_{\boldsymbol{x}} \; h(\boldsymbol{x}), \qquad \text{subject to } f(\boldsymbol{x}) = 0,$$

and assume that we do not know a feasible point. We then consider a sequence of unconstrained minimization problems, where we add to the objective function a penalty function representing the distance from the feasible set:

$$\min_{\boldsymbol{x}} \; h(\boldsymbol{x}) + \lambda f(\boldsymbol{x})^2.$$

Similarly, the penalty function for an inequality $g(\boldsymbol{x}) \leq 0$ would be $\lambda \max\{0, g(\boldsymbol{x})\}^2$. As $\lambda \to \infty$, the solution of the unconstrained problem tends to the solution of the constrained one. Therefore, we can solve the constrained problem by solving a sequence of unconstrained optimization problems, starting from $\lambda = 0$ and progressively raising it. Proper convergence can be proved under appropriate conditions [15].

The *log-barrier* method is useful for dealing with inequalities. Suppose we have to solve the problem

$$\min_{\boldsymbol{x}} \; h(\boldsymbol{x}), \quad \text{subject to } \boldsymbol{x} \leq \overline{\boldsymbol{x}},$$

and assume that we start from a feasible point $\boldsymbol{x}_0 \leq \overline{\boldsymbol{x}}$. Then we solve the sequence of unconstrained optimization problems

$$\min_{\boldsymbol{x}} \; h(\boldsymbol{x}) - \frac{1}{\mu} \sum_i \log\left((\overline{x}_i - x_i)\right).$$

The log term represents a "barrier" that goes to infinity near the bounds. As $\mu \to \infty$, the solution of the unconstrained problem tends to the solution of the constrained one.

3.2 Additional Details

Outliers: We expect that the prior supplied by the user describes most of the environment, but there will always be points that are clearly outside the prior, caused, for example, by clutter in the environment. Therefore, we define another optimization variable, the set INMODEL of points that do respect the prior. Suppose that the likelihood of a point being described by the prior model is $\beta \in (0, 1]$, and, for simplicity, that each point is independent. Then the log-likelihood component $\log(p(\text{INMODEL}))$ can be represented in the cost function by a term $\gamma|\text{INMODEL}|$, with $\gamma = \log(\beta/(1-\beta))$ and $|\text{INMODEL}|$ indicating the number of points.

Upper and lower bounds on $\boldsymbol{\rho}$, $\boldsymbol{\alpha}$: It is possible to derive upper and lower bounds for the variables $\boldsymbol{\alpha}$ and $\boldsymbol{\rho}$. For $\boldsymbol{\rho}$, bounds are obtained by using the initial covariance information. During the optimization, each ρ_i is allowed to vary at most $4\sigma_i$ from the initial estimate $\tilde{\rho}_i$. Because of that, outliers and clutter produce constraints that are impossible to satisfy, and eventually those points are removed from the INMODEL set. As for the normals, it is possible to derive bounds for α_i based on the allowed variation of $\rho_{i-1}, \rho_i, \rho_{i+1}$ and the knowledge of the maximum curvature in the environment.

3.3 Optimization Overview

We rewrite again the form of the optimization problem, with the new variable INMODEL and the bounds on the state.

Problem 4. Find $\boldsymbol{\mathcal{T}}$, INMODEL, $\boldsymbol{x}$ as the solution of:

$$\max_{\boldsymbol{\mathcal{T}}, \boldsymbol{x} \in \text{INMODEL}} \log p(\boldsymbol{x}|\boldsymbol{\rho}) + h_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}) + \gamma|\text{INMODEL}|,$$

$$\text{subject to} \quad f_{\mathcal{T}}(\boldsymbol{x}) = 0, \tag{3}$$
$$g_{\mathcal{T}}(\boldsymbol{x}) \leq 0, \tag{4}$$
$$\underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}. \tag{5}$$

We have to optimize over discrete and continuous variables. The discrete variables are the set INMODEL and the topology $\mathcal{T}$. The continuous variable is $\boldsymbol{x} = (\boldsymbol{\rho}, \boldsymbol{\alpha})$. We solve the problem using two nested levels: the outer level (Algorithm 1) optimizes over INMODEL and $\mathcal{T}$, while the inner level (Algorithm 2) optimizes over $\boldsymbol{x}$, given a particular choice of INMODEL and $\mathcal{T}$. We describe the inner level first.

3.4 Inner Loop: Optimizing x Given INMODEL, $\mathcal{T}$

Algorithm 2 solves Problem 4 assuming that INMODEL, $\mathcal{T}$ have been fixed. We apply a double homotopy transformation to find $\boldsymbol{x}$. We use a penalty function for constraints (3)-(4) and a log-barrier method for constraint (5). Using the log-barrier for the bounds ensures that those are always satisfied during each iteration. Instead, the constraints on the prior are satisfied only in the limit: we start from the measurements and eventually arrive to the surface defined by the prior (Fig. 1).

At each iteration, we take a Newton step with backtracking. All the necessary gradients and Hessians are computed in closed form using symbolic derivations from the user-specified constraints. Moreover, we "convexify" the Hessian if it is not positive-definite by setting negative eigenvalues to a small positive value (Algorithm 2, line 12); this turns the Newton method into gradient descent in the non-convex parts of the state space.

Note that Algorithm 2 might fail to return a feasible point; this will be interpreted by the outer level as a sign that the topology $\mathcal{T}$ is wrong and must be relaxed.

3.5 Outer Loop: Optimizing INMODEL, $\mathcal{T}$

Algorithm 1 optimizes over the set INMODEL and the topology $\mathcal{T}$. Solving this problem exactly has combinatorial complexity, as we would have to try each possible grouping of points into surfaces and regions. To obtain an approximate solution, we use a heuristic approach based on relaxation.

We initialize INMODEL to contain all the points, and $\mathcal{T}$ to result in the strictest set of constraints ($\mathcal{T}_i = \texttt{sameRegion}$). Iteratively, we call the inner level to find a corresponding $\boldsymbol{x}$. If Algorithm 2 finds a feasible $\boldsymbol{x}$, we are done. Otherwise, we try to relax the problem. If the problem is infeasible, some of the prior constraints ($g_{\mathcal{T}}(\boldsymbol{x}) \leq \mathbf{0}, f_{\mathcal{T}}(\boldsymbol{x}) = 0$) are not respected and the corresponding penalty functions are non-zero. We check which couple of nearby points gave the most contribution to the penalty function, and we relax the topology (line 14). If the corresponding $\mathcal{T}_k$ was `sameRegion`, we set it to `differentRegion`; if it was `differentRegion`, we set it to `differentSurface`. We observed that this simple algorithm was effective in finding region and surface boundaries.

In addition, we check whether some regions are too small, and we remove the corresponding points from INMODEL (line 16). This is useful for dealing with outliers.

Algorithm 1. Discrete Optimization of INMODEL, $\mathcal{T}$

```
function [x, T] = map_optimization(ρ̃,Σ_ρ̃,prior):                          1
  % initialize by using all points, and the strictest topology             2
  INMODEL = all; T_k = sameRegion;                                         3
  while True:                                                              4
    [x_,x̄] = geometric_bounds(T,ρ̃,Σ_ρ̃)                                    5
    % Estimate surface normals                                              6
    [α_0, covalpha] = estimate_initial_alpha(ρ̃,Σ_ρ̃,T)                      7
    % Restrict optimization to the INMODEL set                              8
    x_0 = {(ρ̃, α_0)} for i ∈ INMODEL                                       9
    [feasible, x, link_penalties] =                                        10
        inner_optimization(prior,x_0,cov0,T,[x_,x̄])                        11
    if feasible: break                                                     12
    % If not feasible, break the topology based on the penalties           13
    T = break_greedily(T,link_penalties)                                   14
    % Remove points in small regions from the INMODEL set                  15
    [INMODEL,T] = remove_lonely_points(INMODEL,T)                          16
  return [x, T]                                                            17
```

Algorithm 2. Continuous optimization of $\boldsymbol{x}$

```
[feasible,x,link_penalties] = inner_optimization(prior,x_0,Σ_x0,T,[x_,x̄])
  % Obtain functions from prior and topology                               2
  f_T(x), g_T(x), h_T(x) = prior_to_constraints(prior, T)                  3
  for λ=λ_0; λ ≤ λ_max; λ = λ_mult λ:                                      4
    for μ = μ_0; μ ≤ μ_max; μ = μ_mult μ:                                  5
      % return if the point is feasible                                    6
      if f_T(x) < ε: return [true, x]                                      7
      % compute gradient and Hessian of objective + penalties             8
      J(x) = p(x|x_0, Σ_x0) + h_T(x)                                       9
          + log_barrier([x_, x̄],μ,x) + λ penalty(f_T(x), g_T(x))          10
      % convexify the Hessian (do gradient descent if nonconvex)          11
      H = convexify(∇²_x J(x))                                             12
      newton_direction = −inv(H) * ∇_x J(x)                                13
      x = back_tracking(x, newton_direction)                               14
  % the problem is infeasible: compute the penalty for each link          15
  link_penalties = compute_link_penalties(T,x)                             16
  return [false, null, link_penalties]                                     17
```

4 Recovering the Degrees of Freedom

We have shown how to define generic priors (Section 2) and how to perform inference with them (Section 3). We have decoupled the environment prior from the environment representation: while the priors are most general, the representation is always the same. This approach certainly has its advantages in terms of generality and flexibility. However, we lose something with respect to a feature-based approach. The advantages of representing the map with geometric primitives is

that the representation implicitly encodes the *constraints* and *degrees of freedom* of the environment. If we fit a circle to the environment, we implicitly state that 1) the points are constrained to lie on a circle *(*constraints*)*; and 2) the circle can change in radius and position (degrees of freedom). In this section, we show how we can perform a similar analysis even using augmented scans for the representation.

4.1 The Geometric Structure of the Map Space

In equation (1), we let the sensor model depend on the underlying true map "$\boldsymbol{m}$", interpreted as an abstract infinite-dimensional quantity belonging to a certain set $\mathcal{M}$. In order to derive well-grounded results, we have to formalize some intuitive ideas about $\mathcal{M}$ (some of these are commented in more detail elsewhere [2]).

It is intuitive that, for each map $\boldsymbol{m} \in \mathcal{M}$, there will be other elements in $\mathcal{M}$ that have the same *shape* but are rotated/translated to different *poses*. Thus, we can assume that all reasonable sets $\mathcal{M}$ are isomorphic to the product $\mathcal{S} \times \mathrm{SE}(2)$, where $\mathcal{S}$ is called the *shape space*. Given this factorization, we can write an element $\boldsymbol{m} \in \mathcal{M}$ as a couple $\langle \boldsymbol{S}, \boldsymbol{p} \rangle \in \mathcal{S} \times \mathrm{SE}(2)$. This factorization is the basis of many works in the shape-space analysis [8, 9]. Based on that, we introduce a technical condition on the user-defined prior.

Definition 3. *A prior $p(\boldsymbol{m})$ is* pose-independent *if it only depends on the map shape $\boldsymbol{S}$ but not on the map pose $\boldsymbol{p}$:*

$$p(\boldsymbol{m}) = p(\langle \boldsymbol{S}, \boldsymbol{p} \rangle) = p(\boldsymbol{S}).$$

Intuitively, this means that, if the prior allows a certain shape, then it must allow the same shape, rotated, with equal probability; or, equivalently, that observing the environment does not give any information on the robot pose in an external frame. We also state a simple lemma on the ray-tracing function.

Lemma 1. *The observations do not change if robot and map are jointly roto translated: $\boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{p} \rangle, \boldsymbol{q}) = \boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{\delta} \oplus \boldsymbol{p} \rangle, \boldsymbol{\delta} \oplus \boldsymbol{q})$.*

4.2 Analyzing the Degrees of Freedom

Assume we have found a feasible solution $\boldsymbol{x}$. By analyzing the constraints given by the prior, we can recover the degrees of freedom in the solution. More formally, we consider infinitesimal variations $\delta\boldsymbol{x}$ and we examine which ones are allowed by the prior. Recall that $\boldsymbol{x}$ contains both scan readings and surface normals, therefore $\delta\boldsymbol{x}$ belongs to $\mathbb{R}^{2n}$, where n is the number of readings. We first give the mathematical results and then we comment on the derivation.

Proposition 1. *Suppose the prior is pose-independent (Definition 3). Then the space of the allowed variations $\delta\boldsymbol{x}$ to the solution can be factorized as ("$\sqcup$" indicates disjoint union):*

$$\mathbb{R}^{2n} = \mathit{Constr} \sqcup \mathit{Free} = \mathit{Constr} \sqcup (\mathit{Intr} \sqcup \mathit{Extr}),$$

where the subspaces are defined (and computed) as follows:

$$Free \triangleq \ker \nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}), \tag{6}$$

$$Constr \triangleq \mathbb{R}^{2n} - Free, \tag{7}$$

$$Extr \triangleq span\{\nabla_{\boldsymbol{q}} \boldsymbol{r}\}, \tag{8}$$

$$Intr \triangleq Free - Extr. \tag{9}$$

The subspaces Free are the directions corresponding to the map variations allowed by the prior. The subspace Free is further divided in intrinsic *(Intr) degrees of freedom, due to the uncertainty in the map's shape; and* extrinsic *(Extr) degrees of freedom, due to the uncertainty in the map's pose.*

To explain the first division in the subspaces Constr and Free, we just need to consider the equality constraints in the prior, which are represented by the equation $f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}) = 0$. This equation defines a hyper-surface inside $\mathbb{R}^{2n}$ where $\boldsymbol{x}$ is constrained to lie. The tangent plane to this surface is given by directions orthogonal to the gradient $\nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}}$, and corresponds to the (infinitesimal) directions that are allowed by the prior. The subspace Constr is simply the complement of Free.

The further division of Free in intrinsic (Intr) and extrinsic (Extr) degrees of freedom is a more delicate topic. We have seen that the map $\boldsymbol{m}$ can be represented as a couple shape-pose $\langle \boldsymbol{S}, \boldsymbol{p} \rangle$. The subspace Extr identifies the variation in the readings due to the uncertainty in the pose $\boldsymbol{p}$; or, more precisely, due to the uncertain pose between map and sensor. We can state the following result.

Proposition 2. *If the prior is pose-independent, the subspace* $Extr \triangleq span\{\nabla_{\boldsymbol{q}} \boldsymbol{r}\}$ *is contained in Free.*

Proof. Using (2), we write $\boldsymbol{x} = \boldsymbol{r}(\boldsymbol{m}, \boldsymbol{q}) = \boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{p} \rangle, \boldsymbol{q})$. If the prior does not depend on $\boldsymbol{p}$, then $f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}) = 0$ implies

$$f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{\delta} \oplus \boldsymbol{p} \rangle, \boldsymbol{q})) = 0, \qquad \text{for all } \boldsymbol{\delta} \in \mathrm{SE}(2). \tag{10}$$

Given Lemma 1, we obtain that $f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{p} \rangle, \ominus \boldsymbol{\delta} \oplus \boldsymbol{q})) = 0$, for all $\boldsymbol{\delta} \in \mathrm{SE}(2)$. This means that

$$f_{\boldsymbol{\mathcal{T}}}(\boldsymbol{r}(\langle \boldsymbol{S}, \boldsymbol{p} \rangle, \boldsymbol{q})) = 0, \qquad \text{for all } \boldsymbol{q} \in \mathrm{SE}(2).$$

Intuitively, this says that the updated readings still respect the prior no matter where the robot is placed in the environment. If a function ($f_{\boldsymbol{\mathcal{T}}}$) is constant with respect to an argument ($\boldsymbol{q}$), the derivative with respect to that argument is 0. In our case, using the chain rule, we obtain:

$$\nabla_{\boldsymbol{q}} f_{\boldsymbol{\mathcal{T}}} = \nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}} \cdot \nabla_{\boldsymbol{q}} \boldsymbol{r} = \boldsymbol{0}.$$

Therefore, $\nabla_{\boldsymbol{q}} \boldsymbol{r}$ is always orthogonal to $\nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}}$; that is, $\mathrm{span}\{\nabla_{\boldsymbol{q}} \boldsymbol{r}\} \subset \ker \nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}} =$ Free.

Extr = span $\{\nabla_{\boldsymbol{q}} \boldsymbol{r}\}$ are the possible variations in the measurements due to the sensor movement. By contrast, the directions in Intr are due to the variation in the map shape $\boldsymbol{S}$, and correspond to the intuitive notion of the degrees of freedom in the structure. To compute them, we use equations (6), (8), (9). Note that we used the concept of map factorization in $\boldsymbol{m} = \langle \boldsymbol{S}, \boldsymbol{p} \rangle$ only as a theoretical tool in deriving the results. In practice, we do not need to know anything about such abstract representation; the only quantities we have to compute are $\nabla_{\boldsymbol{q}} \boldsymbol{r}$ and $\nabla_{\boldsymbol{x}} f_{\boldsymbol{\mathcal{T}}}$, which lie in the very concrete measurement space. The procedure is completely automatic and allows to recover the degrees of freedom for any prior.

4.3 Covariance Shrinking

Other than for visualization purposes, we can use the degrees of freedom knowledge for computing the posterior uncertainty of the estimate. Assume that the covariance of the initial estimate $\boldsymbol{x}_0$ was $\boldsymbol{\Sigma}_{\boldsymbol{x}_0}$. If the prior has only constraints and not energies (i.e., there is no term $h_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x})$), we can obtain the posterior covariance $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ simply by projecting $\boldsymbol{\Sigma}_{\boldsymbol{x}_0}$ onto the subspace Free. Let the matrix P_{Free} be a projector onto Free. Then the posterior covariance estimate is $\boldsymbol{\Sigma}_{\boldsymbol{x}} = P_{\text{Free}} \boldsymbol{\Sigma}_{\boldsymbol{x}_0} P_{\text{Free}}^T$. If there is a term $h_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x})$, we have to account for the further reduction of uncertainty. Treating it as an additional observation, we obtain that $\boldsymbol{\Sigma}_{\boldsymbol{x}} = P_{\text{Free}} (\boldsymbol{\Sigma}_{\boldsymbol{x}_0}^{-1} + \nabla^2 h_{\boldsymbol{\mathcal{T}}}^{-1})^{-1} P_{\text{Free}}^T$. Similarly, one can recover the contribution to $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ due to extrinsic or intrinsic uncertainty by projecting onto Extr or Intr. This process has a solid geometric intuition; see also, for example, Chapter 3 of Paul Newman's thesis [10]. The reader should note that this linearized analysis has the usual limitations [5].

For priors with many constraints, the rank of $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ is very low. Thus, it is better to represent it by its non-null eigensystem, which can be interpreted as the allowed scan eigenvariations $\{\langle \boldsymbol{v}^m, \sigma_m \rangle\}_{m=1}^{\dim(\text{Free})}$, each representing a direction $\boldsymbol{v}^m = \langle \boldsymbol{\rho}^m, \boldsymbol{\alpha}^m \rangle$ and corresponding uncertainty σ_m in that direction.

5 Experiments

We have conducted experiments with both synthetic and real data.

Fig. 3 shows an example test case, with a simulated scan from a square environment, using the rectangular prior. The original noisy simulated data is depicted in Fig 3(a), whereas Fig 3(b) presents the corrected measurements and the proper division of the scan into different regions. Fig 4 shows the orientation angles for all the readings corrected by applying our method (x-like crosses), with the ground-truth represented by dots and the initial estimates represented by circles. The vertical crosses indicate the bounds. The solution gets so close to the ground-truth that they can hardly be distinguished in the plot, with an average error of 0.26° for several tests and the walls being well aligned. We obtain similar results with a variety of other simulated environments. We also

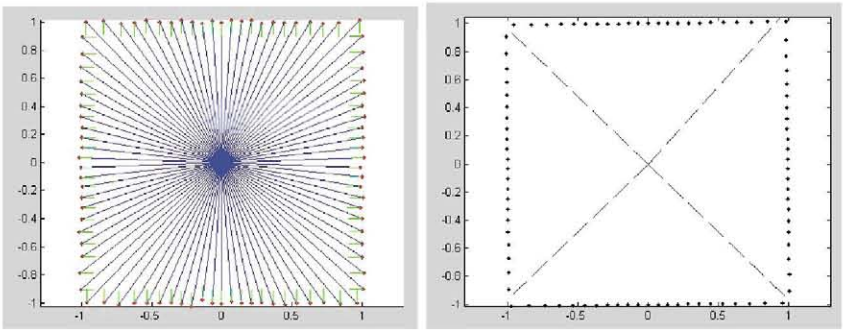


Fig. 3. (a): Noisy simulated scan. (b): Corrected measurements and topology $\mathcal{T}$ (dashed edges for different regions).

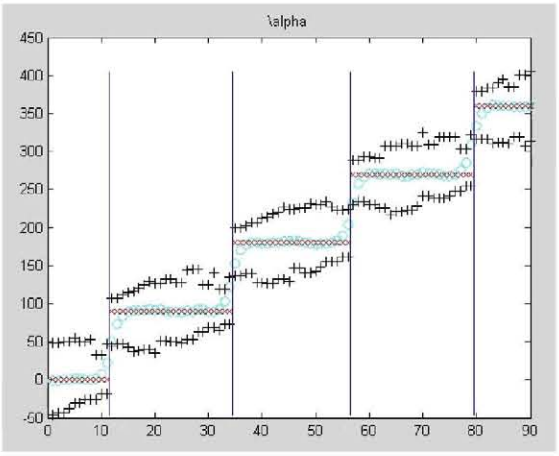


Fig. 4. Corrected orientation angles (x-like symbols), ground truth (small dots), initial values (circles) and extracted topology $\mathcal{T}$ (vertical edges for different regions)

conducted tests with a random number of outliers, Fig. 5 shows some examples. Other experiments had similar results, they are not presented here for lack of space. The quantitative analysis of these results requires some further work. It is not always easy to assess whether a measurement around the corner is properly considered an outlier or whether a point introduced as an outlier is an outlier indeed, it depends on its neighbors, on how the random outliers are distributed around the scan.

The whole process described in Section 3 can be seen in action with real data from a Hokuyo laser sensor in Fig. 6. Even if most of the environment is polygonal, the regions of the polygonal surfaces are interrupted by random

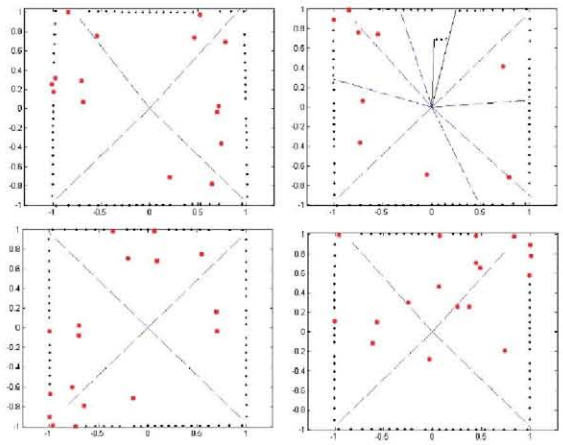
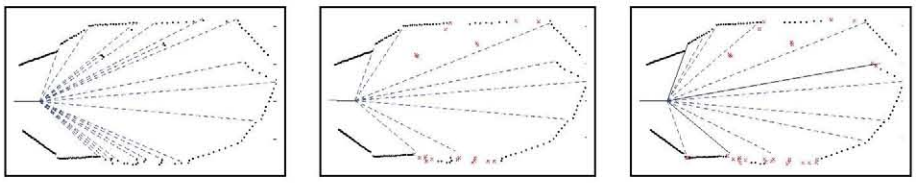


Fig. 5. Output of some experiments with outliers. Identified outliers, depicted as bigger dots, were eliminated from INMODEL



(a) Topology $\mathcal{T}$ after the first iteration (b) Intermediate iteration (c) Final topology

Fig. 6. The outer level optimization (Algorithm 1) works on the discrete variables, deciding which sensor readings can be described by the prior (variable INMODEL) and the division in regions/surfaces (variable $\mathcal{T}$). The pictures show the evolution of the topology. Solid lines indicate boundaries between surfaces; dashed lines indicate borders between regions. Dark crosses indicate readings outside the INMODEL set. (a): Some decisions on $\mathcal{T}$ can be taken based on the geometric constraints and the knowledge of the maximum curvature, specified in the prior. (b): The rest of the algorithm guesses where the region/surfaces boundaries are based on a greedy relaxation algorithm. Clutter and outliers tend to be isolated in small regions that are later removed. (c): The final result is feasible according to the prior; the outliers have been removed from the set INMODEL.

clutter and outliers (Fig. 6a). The first part of the relaxation introduces several breaks around outliers (Fig. 6b) producing a very fragmented topology. Then we remove the clutter from INMODEL and we can return to the simple correct topology for the rest of the points (Fig.6c).

Fig. 7 shows some more experiments with real data acquired at Principe Felipe Science Museum in Spain. Most outliers come from glass panels and people.

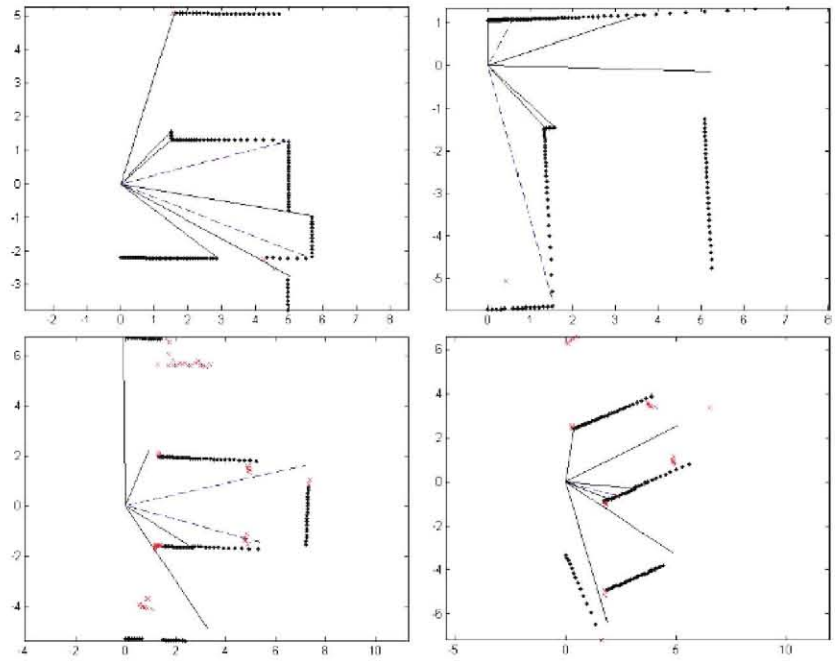
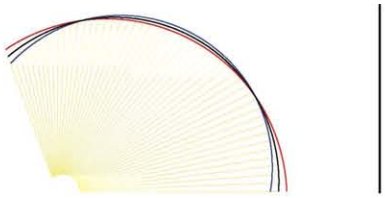


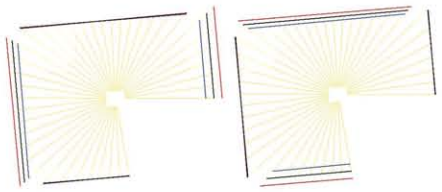
Fig. 7. More experiments with real scans. Data from Principe Felipe Museum (Valencia, Spain), highest floor

Regarding the degrees of freedom extraction, the system recognizes well, for instance, that a circular environment has only one intrinsic degree of freedom (the radius of the circle) (Fig. 8a). A rectangular environment has two degrees of freedom (Fig. 8b) using the rectangular prior (Fig. 2b), but 5 if we use the polygonal prior (Fig. 2a), because the walls orientation is not constrained. Our system recognizes well the degrees of freedom in more complicated situations. For example, an environment with two circles has three degrees of freedom (the radii and the distance between the centers); however, they quickly become hard to visualize. Moreover, in the figures we plot only the variation of the readings because the variations of the normals are hard to visualize as well.

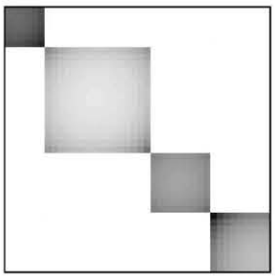
Fig. 8 also shows an example of covariance shrinkage with a rectangular environment. We assume that the initial covariance of $\tilde{\boldsymbol{\rho}}$ is band-diagonal with slight correlation across neighbours. After the projection, the posterior covariance (Fig. 8c) correlates readings corresponding to the same surface or region. As shown in Fig. 8d, there is a dramatic uncertainty reduction.



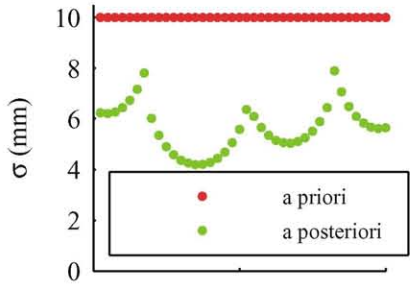
(a) Eigenvariation for circular environment with circular prior (Fig. 2e)



(b) Eigenvariations for rectangular environment with rectangular prior (Fig. 2b)



(c) A-posteriori readings covariance



(d) Readings standard deviations

Fig. 8. After we have found the solution to the optimization problem, the inference engine uses the knowledge of the prior for extracting the intrinsic degrees of freedom (*scan eigenvariations*) and for *shrinking* the covariance by projecting it onto the Free subspace. (a): For example, the inference engine can recognize that a circular environment has one allowed scan eigenvariations. (b): In the case of a rectangular environment and rectangular prior (Fig. 2b), we find 2 allowed scan eigenvariations. These can be interpreted as the variations of width and height of the environment. (c): We can shrink the a-priori readings covariance by projecting it onto the constraints. In this case, we assume that the a-priori covariance (not shown) has slight correlation between consecutive readings. The a-posteriori covariance has very low rank, and distant readings become correlated because of the structure. (d): The shrinking can be visualized by plotting the diagonal elements of a-priori and a-posteriori covariance.

6 Conclusions and Future Work

We have shown how to build an inference engine that can use different priors with the same representation. The priors are defined by the user in a domain-specific language. The problem of approximating the map posterior is turned into a constrained optimization problem and a covariance projection over the unconstrained directions. This way, it is possible to apply structured priors

(polygonal, rectangular, etc.) using a unified representation. Besides being able to reason in terms of regions and surfaces, one can recover the "structure" information under the form of scan eigenvariations, using the degrees-of-freedom analysis.

As part of future work, we plan to improve the greedy Algorithm 1 by introducing backtracking. The incorporation of automatic methods for the representation of the prior of an environment could be another future contribution. We are also interested in testing how preprocessing different sensor data with our method may help scan matching standard techniques. Finally, we are working on the integration of this algorithm into complete SLAM methods, by using the reduced degrees of freedom for global map optimization.

References

1. Beevers, K., Huang, W.: Inferring and Enforcing Relative Constraints in SLAM. In: Algorithmic Foundation of Robotics VII. Springer, Heidelberg (2008)
2. Censi, A.: On achievable accuracy for pose tracking (2009), http://purl.org/censi/2008/posetracking
3. Chong, K., Kleeman, L.: Sonar based map building for a mobile robot (1997)
4. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with Rao-Blackwellized particle filters 23(1), 34–46 (2007)
5. Julier, S., LaViola, J.: On Kalman filtering with nonlinear equality constraints 55(6) (2007)
6. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
7. Kümmerle, R., Steder, B., Dornhege, C., Kleiner, A., Grisetti, G., Burgard, W.: Large scale graph-based SLAM using aerial images as prior information. Journal of Autonomous Robots 30(1), 25–39 (2011)
8. Le, H., Kendall, D.G.: The Riemannian structure of Euclidean shape spaces: A novel environment for statistics. Annals of Statistics 21(3), 1225–1271 (1993)
9. Michor, P.W., Mumford, D.: Riemannian geometries on spaces of plane curves. J. of the European Math. Soc. 8, 1–48 (2004)
10. Newman, P.: On the Structure and Solution of the Simultaneous Localisation and Map Building Problem. Ph.D. thesis, U. of Sydney (1999)
11. Nguyen, V., Harati, A., Siegwart, R.: A lightweight SLAM algorithm using orthogonal planes for indoor mobile robotics (2007)
12. Parsley, M.P., Julier, S.J.: Towards the exploitation of prior information in SLAM. In: Proc. of the IEEE Int. Conf. on Intelligent Robots and Systems, IROS (2010)
13. Plagemann, C., Kersting, K., Pfaff, P., Burgard, W.: Gaussian beam processes: A nonparametric bayesian measurement model for range finders. In: Robotics: Science and Systems, RSS (2007)
14. Rodríguez-Losada, D., Matía, F., Jiménez, A., Galán, R.: Consistency improvement for SLAM-EKF for indoor environments (2006)
15. Ruszczynski, A.P.: Nonlinear Optimization. Princeton U. Press (2006)
16. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press (2005)
17. Trevor, A.J.B., Rogers, J.G., Nieto, C., Christensen, H.I.: Applying domain knowledge to SLAM using virtual measurements. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA (2010)